

BEST AVAILABLE COPY

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2003-029969

(43)Date of publication of application : 31.01.2003

(51)Int.Cl.

G06F 9/42
G06F 9/54
H03K 19/173

(21)Application number : 2002-060515

(71)Applicant : TOKYO ELECTRON DEVICE LTD
NISHIHARA AKINORI

(22)Date of filing : 06.03.2002

(72)Inventor : NISHIHARA AKINORI
HASEBE TETSUYA
HAYASHI HIROAKI
MITA TAKASHI

(30) Priority

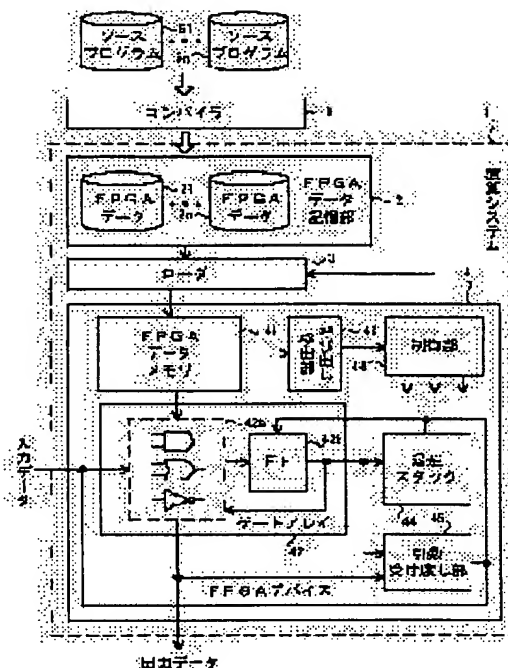
Priority number : 2001139951 Priority date : 10.05.2001 Priority country : JP

(54) ARITHMETIC SYSTEM

(57)Abstract:

PROBLEM TO BE SOLVED: To directly realize the performance of a large scale program composed of a plurality of program modules by hardware without using a general purpose CPU.

SOLUTION: For a gate array 42, logical constitution among gate circuits 42a is attained according to an FPGA data module stored in an FPGA data memory 41 and an arithmetic operation is performed in terms of the hardware. When it is detected that the module stored in the FPGA data memory 41 is the one calling the other module by a calling detection part 43, the data of the halfway result of the arithmetic operation held in a flip-flop 42b are saved in a saving stack 44 and an argument to be delivered to the module of a calling destination is temporarily preserved in an argument receipt and delivery part 45. Thereafter, the module of the calling destination is loaded to a loader 3, and at the time of returning from the arithmetic operation by the module of the calling destination to the module of a calling origin, the data saved in the saving stack 44 are written back to the flip-flop 42b.



LEGAL STATUS

[Date of request for examination]

21.08.2002

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

【特許請求の範囲】

【請求項 1】複数のプログラムモジュールからなるプログラムを記憶するプログラム記憶手段と、

前記プログラム記憶手段に記憶されたプログラムモジュールをメモリにロードするロード手段と、

複数の論理回路を含み、前記ロード手段によってメモリにロードされたプログラムモジュール中の命令に従った信号を前記複数の論理回路の 1 以上に入力することで、該ロードされたプログラムモジュールに応じた演算を実行する論理演算手段と、

前記論理演算手段の内部状態を退避する退避手段と、所定の条件が成立した場合に、前記論理演算手段の内部状態を前記退避手段に退避すると共に、他のプログラムモジュールを前記ロード手段にロードさせ、該他のプログラムモジュールに応じた演算の実行を終了した後に前記退避手段に退避した内部状態を前記論理演算手段に戻してから、元のプログラムモジュールに応じた演算の実行に復帰させる制御手段とを備えることを特徴とする演算システム。

【請求項 2】前記複数のプログラムモジュールのうちの少なくとも一部のプログラムモジュールは、他のプログラムモジュールを呼び出す機能を含み、前記論理演算手段で演算を実行しているプログラムモジュール中の命令における他のプログラムモジュールの呼び出しを検出する呼び出し検出手段をさらに備え、前記制御手段は、前記呼び出し検出手段が他のプログラムモジュールの呼び出しを検出した場合に、前記論理演算手段の内部状態を前記退避手段に退避すると共に、呼び出し先のプログラムモジュールを前記ロード手段にロードさせ、呼び出し先のプログラムモジュールに応じた演算の実行を終了した後に前記退避手段に退避した内部状態を前記論理演算手段に戻してから、呼び出し元のプログラムモジュールに応じた演算の実行に復帰させることを特徴とする請求項 1 に記載の演算システム。

【請求項 3】前記制御手段によって実行が切り替えられるプログラムモジュール間において引数を受け渡すための引数受け渡し手段をさらに備えることを特徴とする請求項 2 に記載の演算システム。

【請求項 4】前記退避手段は、先入れ後出し方式のスタックによって構成されることを特徴とする請求項 1 乃至 3 のいずれか 1 項に記載の演算システム。

【請求項 5】自己に供給された第 1 のプログラムモジュールをメモリにロードするロード手段と、複数の論理回路を含み、前記ロード手段によってメモリにロードされた前記第 1 のプログラムモジュール中の命令に従った信号を前記複数の論理回路の 1 以上に入力することで、ロードされた当該第 1 のプログラムモジュールに応じた演算を実行する論理演算手段と、

前記論理演算手段の内部状態を退避する退避手段と、所定の条件が成立した場合に、自己に着脱可能に接続さ

れた外部の他の演算システムに第 2 のプログラムモジュールをロードさせ、当該他の演算システムが当該第 2 のプログラムモジュールに応じた演算の実行を終了し、演算結果を自己に供給した後に、前記論理演算手段を前記第 1 のプログラムモジュールに応じた演算の実行に復帰させる制御手段とを備えることを特徴とする演算システム。

【請求項 6】複数のプログラムモジュールからなるプログラムを記憶し、当該プログラムモジュールを前記ロード手段に供給するプログラム記憶手段を備えることを特徴とする請求項 5 に記載の演算システム。

【請求項 7】前記第 1 のプログラムモジュールは、前記第 2 のプログラムモジュールを呼び出す機能を含み、前記論理演算手段が演算を実行している前記第 1 のプログラムモジュール中の命令における前記第 2 のプログラムモジュールの呼び出しを検出する呼び出し検出手段をさらに備え、

前記制御手段は、前記呼び出し検出手段が前記第 2 のプログラムモジュールの呼び出しを検出した場合に、第 2 のプログラムモジュールを外部の他の演算システムにロードさせ、当該他の演算システムが当該第 2 のプログラムモジュールに応じた演算の実行を終了し、演算結果を自己に供給した後に、前記論理演算手段を前記第 1 のプログラムモジュールに応じた演算の実行に復帰させることを特徴とする請求項 5 又は 6 に記載の演算システム。

【請求項 8】前記プログラム記憶手段に記憶された各プログラムモジュール中の命令は、前記論理演算手段を構成する論理回路に入力する信号に応じたコードによって構成されていることを特徴とする請求項 1 乃至 7 のいずれか 1 項に記載の演算システム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、プログラムの実行をハードウェアで直接的に実現できる演算システムに関し、特に大規模プログラムの実行に適した演算システムに関する。

【0002】

【従来の技術】現在の汎用コンピュータは、CPU (Central Processing Unit) がメモリに記憶されたプログラム中の命令を順次解釈しながら、演算を進めていく。CPU は、プログラムで実行すべき演算をソフトウェアで実現するものであり、必ずしもその演算に対して最適なハードウェア構成となっていないため、最終的な演算結果を得るまでに多くのオーバーヘッドが存在する。

【0003】これに対して、プログラムの実行をハードウェアで直接的に実現するための技術として、例えば、特表平 8-504285 号公報 (国際公開 WO 94/10627 号公報) や特表 2000-516418 号公報 (国際公開 WO 98/08306 号公報) に示されているような、フィールドプログラマブルゲートアレイ (F

P G A) を利用した演算システムが知られている。

【0004】F P G A は、プログラムとして論理データを与えることで論理回路間の結線論理を変更し、これによってハードウェア的に演算結果を得ることをできるようにしたものである。F P G A を利用して演算を行うことによって、特定の演算専用構成されたハードウェア回路ほどは高速ではないが、従来の汎用コンピュータのような C P U による演算に比べると、非常に高速で演算結果を得ることができる。

【0005】

【発明が解決しようとする課題】ところで、現在の汎用コンピュータで実行されているプログラム、特に大規模なプログラムは、複数のモジュールに分割して作成されている。そして、1 のプログラムモジュールが他のプログラムモジュールを呼び出しながら、全体としてのプログラムの実行を進めていくようになっている。こうしてプログラムモジュール別に開発を進めたり、各プログラムモジュールを部品として利用したりすることにより、プログラムの開発期間を短縮することができる。

【0006】しかしながら、上記した従来の F P G A を用いた演算システムでは、ハードウェアとしてのモジュール分割は考えられていても、ソフトウェアとしてのモジュール分割は考えられていなかった。つまり、ソフトウェアとして1のプログラムモジュールから他のプログラムモジュールを呼び出し、呼び出したプログラムモジュールの実行を終了した後、元のプログラムモジュールに復帰するというように、複数のプログラムモジュールを適時実行していくことにより大規模プログラムの実行を可能とする仕組みは考えられていなかった。

【0007】このため、従来の F P G A を用いた演算システムで実行可能なプログラムは、実質的に1のみのモジュールで作成されたプログラムでなくてはならないという制約があった。つまり、大規模プログラムの実行が事実上不可能で、その適用範囲は限られるという問題があった。

【0008】本発明は、上記した従来技術の問題点を解消するためになされたものであり、汎用の C P U を用いることなく、複数のプログラムモジュールからなる大規模プログラムの実行をハードウェアで直接的に実現した演算システムを提供することを目的とする。

【0009】

【課題を解決するための手段】上記目的を達成するため、本発明の第1の観点に係る演算システムは、複数のプログラムモジュールからなるプログラムを記憶するプログラム記憶手段と、前記プログラム記憶手段に記憶されたプログラムモジュールをメモリにロードするロード手段と、複数の論理回路を含み、前記ロード手段によってメモリにロードされたプログラムモジュール中の命令に従った信号を前記複数の論理回路の1以上に入力することで、該ロードされたプログラムモジュールに応じた

演算を実行する論理演算手段と、前記論理演算手段の内部状態を退避する退避手段と、所定の条件が成立した場合に、前記論理演算手段の内部状態を前記退避手段に退避すると共に、他のプログラムモジュールを前記ロード手段にロードさせ、該他のプログラムモジュールに応じた演算の実行を終了した後に前記退避手段に退避した内部状態を前記論理演算手段に戻してから、元のプログラムモジュールに応じた演算の実行に復帰させる制御手段とを備えることを特徴とする。

10 【0010】上記演算システムでは、メモリにロードするプログラムモジュールを切り替えるときに、論理演算手段の内部状態を退避し、また、復元する仕組みを備えている。このため、複数のプログラムモジュールからなる大規模なプログラムも、メモリにロードするプログラムを切り替え、論理演算手段中の論理回路に入力する信号を変えていくことで、ハードウェア的に高速に演算を実行していくことができる。

【0011】上記演算システムにおいて、前記複数のプログラムモジュールのうちの少なくとも一部のプログラムモジュールは、他のプログラムモジュールを呼び出す機能を含むものであってもよい。この場合において、上記演算システムは、前記論理演算手段で演算を実行しているプログラムモジュール中の命令における他のプログラムモジュールの呼び出しを検出する呼び出し検出手段をさらに備えるものとしてでき、前記制御手段は、前記呼び出し検出手段が他のプログラムモジュールの呼び出しを検出した場合に、前記論理演算手段の内部状態を前記退避手段に退避すると共に、呼び出し先のプログラムモジュールを前記ロード手段にロードさせ、呼び出し先のプログラムモジュールに応じた演算の実行を終了した後に前記退避手段に退避した内部状態を前記論理演算手段に戻してから、呼び出し元のプログラムモジュールに応じた演算の実行に復帰させるものとしてできる。

【0012】ここで、前記制御手段によって実行が切り替えられるプログラムモジュール間において引数を受け渡すための引数受け渡し手段をさらに備えていてもよい。

【0013】これらの仕組みをさらに備えることによって、モジュールの呼び出しを含む大規模なプログラムをハードウェア的に高速に実行することが可能となる。

【0014】上記演算システムにおいて、前記退避手段は、先入れ後出し方式のスタックによって構成されたものとしてできる。

【0015】このようなスタックで構成される退避手段により、例えば、別のプログラムモジュールから呼び出されたプログラムモジュールが、さらに他のプログラムモジュールを呼び出すといった処理も可能となる。また、あるプログラムモジュールが、そのプログラムモジュール自身を呼び出す再帰型のプログラムを実行するこ

とも可能となる。

【0016】また、本発明の第2の観点に係る演算システムは、自己に供給された第1のプログラムモジュールをメモリにロードするロード手段と、複数の論理回路を含み、前記ロード手段によってメモリにロードされた前記第1のプログラムモジュール中の命令に従った信号を前記複数の論理回路の1以上に入力することで、ロードされた当該第1のプログラムモジュールに応じた演算を実行する論理演算手段と、前記論理演算手段の内部状態を退避する退避手段と、所定の条件が成立した場合に、自己に着脱可能に接続された外部の他の演算システムに第2のプログラムモジュールをロードさせ、当該他の演算システムが当該第2のプログラムモジュールに応じた演算の実行を終了し、演算結果を自己に供給した後に、前記論理演算手段を前記第1のプログラムモジュールに応じた演算の実行に復帰させる制御手段とを備えることを特徴とする。

【0017】上記演算システムは、第2のプログラムモジュールが表す演算へと処理を切り替えるときに、外部の他の演算システムに第2のプログラムモジュールをロードさせる構成を備えている。このため、複数のプログラムモジュールからなる大規模なプログラムも、単一の演算システムでは短時間で完了できない演算や、並列処理を要する演算も、ハードウェア的に高速に実行していくことができる。また、3個以上の演算システムを連鎖的に接続することも可能であるから、演算の手順を柔軟に構成することが可能である。

【0018】上記演算システムは、たとえば、複数のプログラムモジュールからなるプログラムを記憶し、当該プログラムモジュールを前記ロード手段に供給するプログラム記憶手段を備えることにより、ロード手段にロードさせるプログラムモジュールを確保する。

【0019】上記演算システムにおいて、前記第1のプログラムモジュールは、前記第2のプログラムモジュールを呼び出す機能を含むものであってもよい。この場合において、上記演算システムは、前記論理演算手段が演算を実行している前記第1のプログラムモジュール中の命令における前記第2のプログラムモジュールの呼び出しを検出する呼び出し検出手段をさらに備えるものとして、前記制御手段は、前記呼び出し検出手段が前記第2のプログラムモジュールの呼び出しを検出した場合に、第2のプログラムモジュールを外部の他の演算システムにロードさせ、当該他の演算システムが当該第2のプログラムモジュールに応じた演算の実行を終了し、演算結果を自己に供給した後に、前記論理演算手段を前記第1のプログラムモジュールに応じた演算の実行に復帰させるものとして行うことができる。

【0020】上記演算システムにおいて、前記プログラム記憶手段に記憶された各プログラムモジュール中の命令は、前記論理演算手段を構成する論理回路に入力する

信号に応じたコードによって構成されたものであってもよい。

【0021】なお、各プログラムモジュール中の命令を構成するコードは、ハードウェア記述が可能な言語で記述されたソースプログラムをコンパイルすることによって得ることができる。この場合、モジュール別にソースプログラムを開発したり、モジュールの部品としての利用が可能となり、プログラムの開発期間を短縮することが可能となる。

10 【0022】

【発明の実施の形態】以下、添付図面を参照して、本発明の実施の形態について説明する。

【0023】図1は、この実施の形態にかかる演算システムの構成を示すブロック図である。図示するように、この演算システム1は、FPGAデータ記憶部2と、ロード3と、FPGAデバイス4とから構成されている。FPGAデータ記憶部2には、複数のモジュールに分かれたFPGAデータモジュール21～2nを記憶している。

20 【0024】FPGAデータモジュール21～2nは、それぞれハードウェア記述が可能なプログラム言語で記述されている複数のモジュールに分かれたソースプログラム51～5nを、FPGAデバイス4の論理記述を行うべくコンパイラ6がコンパイルしたモジュール毎のデータである。ソースプログラム51～5nのうちの少なくとも1のモジュールは、他のモジュールのソースプログラム51～5nを呼び出す機能を含んでおり、FPGAデータモジュール21～2nには、他のモジュールの呼び出しのためのデータも含まれている。

30 【0025】ロード3は、論理回路等より構成されており、FPGAデータ記憶部2に記憶されたFPGAデータモジュール21～2nをモジュール単位でFPGAデバイス4に適時ロードする。ロード3によるFPGAデータモジュール21～2nのロードの指示は、演算の実行の開始時に外部から与えられる他、FPGAデバイス4による演算の実行によっても与えられる。

40 【0026】FPGAデバイス4は、ロード3によってロードされたFPGAデータモジュール21～2nに従って論理構成を行い、外部からの入力データに所定の演算を施して出力データとして出力するもので、FPGAデータメモリ41と、ゲートアレイ42と、呼び出し検出部43と、退避スタック44と、引数受け渡し部45と、制御部46とを備えている。呼び出し検出部43、退避スタック44、引数受け渡し部45及び制御部46は、論理回路等より構成されている。

50 【0027】FPGAデータメモリ41は、RAM(Random Access Memory)によって構成され、ロード3がロードしたFPGAデータモジュールを記憶する。ゲートアレイ42は、AND、OR、NOTなどの複数のゲート回路42aと、演算の途中結果を内部状態として保持

している複数のフリップフロップ 42b とを含んでいる。各ゲート回路 42a の出力論理は、FPGA データメモリ 41 に記憶された FPGA データモジュールに従って変更される。また、各フリップフロップ 42b は、所望のデータを外部から書き込むことができるようになっている。

【0028】呼び出し検出部 43 は、FPGA データメモリ 41 に記憶された FPGA データモジュールに含まれる他のモジュールの呼び出しのためのデータを検出する。退避スタック 44 は、呼び出し検出部 43 によって他のモジュールの呼び出しのためのデータが検出されたとき、ゲートアレイ 42 中のフリップフロップ 42b に保持されているデータと、呼び出し元の FPGA データモジュールの識別データとを、先入れ後出し方式で退避するためのスタックである。

【0029】引数受け渡し部 45 は、モジュールの呼び出し、復帰の際において呼び出し元と呼び出し先の FPGA データモジュール間における引数の受け渡しを行うものである。より詳細に説明すると、呼び出しの際には、呼び出し元の FPGA データモジュールに従った演算の途中結果としてフリップフロップ 42b の所定のものに保持されていたデータを、呼び出し先の FPGA データモジュールに従った演算の入力（引数）として与える。復帰の際には、呼び出し先の FPGA データモジュールに従った演算結果（戻り値）の出力データを、ゲートアレイ 42 中のフリップフロップ 42b の所定のものに書き込む。

【0030】制御部 46 は、呼び出し検出部 43 が他のモジュールの呼び出しのためのデータを検出した場合、当該呼び出しのためのデータの直前までの FPGA データモジュールに従った演算の途中結果としてフリップフロップ 42b のそれぞれに保持されているデータと、呼び出し元のデータモジュールの識別データとを退避スタック 44 に退避させると共に、呼び出し先の FPGA データモジュールに従った演算で使用するデータを保持するフリップフロップ 42b のデータを、引数受け渡し部 45 に一時保持させる。その後、呼び出し先の FPGA データモジュールをロード 3 にロードさせ、引数受け渡し部 45 に一時保持したデータをゲートアレイ 42 に入力データとして与える。

【0031】制御部 46 は、また、呼び出された FPGA データモジュールに従った演算が終了したときに、その出力データを引数受け渡し部 45 に一時保持させる。その後、退避スタック 44 に退避された呼び出し元のデータモジュールの識別データに従ってロード 3 に FPGA データモジュールをロードさせ、退避スタック 44 に退避されたデータをフリップフロップ 42b に復帰させると共に、引数受け渡し部 45 に一時保持させたデータをフリップフロップ 42b の所定のものに書き込ませる。

【0032】なお、FPGA デバイス 4 に外部から入力される入力データは、キーボードなどの入力装置から入力されるデータその他、磁気ディスク装置などの外部記憶装置から読み出されたデータであってもよい。また、FPGA デバイス 4 から外部に出力される出力データは、ディスプレイ装置などの出力装置から出力する他、外部記憶装置に書き込むものであってもよく、さらに、周辺機器を制御するための制御データであってもよい。

【0033】以下、この実施の形態にかかる演算システムにおける動作について、具体的な例に基づいて説明する。ここでは、FPGA データモジュール 21 が最初にロードされるものとし、FPGA データモジュール 21 は、FPGA データモジュール 2n を呼び出すものとする。

【0034】FPGA データモジュール 2 が FPGA データメモリ 41 にロードされると、これに従ったレベルの信号がゲート回路 42a に入力され、ゲートアレイ 42 を構成するゲート回路 42a が論理構成される。そして、ゲートアレイ 42 に外部からの入力データが入力されることによって、FPGA データモジュール 21 に応じた演算がゲートアレイ 42 において実行される。

【0035】一方、呼び出し検出部 43 は、FPGA データメモリ 41 にロードされた FPGA データモジュール 21 に FPGA データモジュール 2n を呼び出すためのデータが含まれていることを検出し、その旨を制御部 46 に通知する。制御部 46 は、その呼び出しにかかる部分の直前までの演算の途中結果としてフリップフロップ 42b に保持されているデータ（ゲートアレイ 42 の内部状態）を、呼び出し元の FPGA データモジュール 21 を識別するためのデータと共に退避スタック 44 の一番上に退避させる。また、フリップフロップ 42b に保持されているデータのうちの呼び出し先の FPGA データモジュール 2n に引数として渡すものを、引数受け渡し部 45 に一時保存させる。

【0036】その後、制御部 46 は、ロード 3 を制御し、呼び出し先である FPGA データモジュール 2n を FPGA データメモリ 41 にロードさせる。FPGA データモジュール 2n がロードされると、これに従ったレベルの信号がゲート回路 42a に入力され、ゲートアレイ 42 を構成するゲート回路 42a が論理構成される。また、引数受け渡し部 45 に引数として一時保存されたデータが、入力データとしてゲートアレイ 42 に入力され、FPGA データモジュール 2n に応じた演算がゲートアレイ 42 において実行される。

【0037】この演算が終了すると、制御部 46 は、ゲートアレイ 42 からの出力データを呼び出し元の FPGA データモジュール 21 に渡す引数として引数受け渡し部 45 に一時保存させる。制御部 46 は、さらに退避スタック 44 の一番上に退避されたデータを参照することでロード 3 を制御し、呼び出し元の FPGA データモジ

ジュール 21 を F P G A データメモリ 41 に再びロードさせる。

【0038】呼び出し元の F P G A データモジュール 21 が再びロードされると、制御部 46 は、退避スタック 44 の一番上に退避されていた内部状態のデータをフリップフロップ 42 b のそれぞれに書き戻し、ゲートアレイ 42 の内部状態を復元させる。さらに、引数受け渡し部 45 に引数として一時保存されていたデータをフリップフロップ 42 b の所定のものに書き込む。この状態でゲートアレイ 42 において F P G A データモジュール 21 に従った演算が再開され、最終的な演算結果が出力データとして出力されることとなる。

【0039】なお、F P G A データモジュール 21 から呼び出された F P G A データモジュール 2 n が、さらに他の F P G A データモジュールを呼び出すものであっても演算を実行することができる。F P G A データモジュール 2 n がさらに他のモジュールを呼び出すことを呼び出し検出部 43 が検出した場合にも、制御部 46 は、上記と同じような制御を行うものとすればよい。

【0040】以上説明したように、この実施の形態にかかる演算システムでは、ゲートアレイ 42 の内部状態（フリップフロップ 42 b が保持するデータ）を退避スタック 44 に退避した後に、ロード 3 は、実行中のモジュールとは異なる F P G A データモジュールを F P G A データメモリ 41 にロードするようにしている。また、退避スタック 44 に退避した状態をゲートアレイ 42 に復元してから元のモジュールに復帰することができるようになっている。このため、各 F P G A データモジュールを F P G A データメモリ 41 に適時ロードしていくことによって、複数のモジュールからなる大規模なプログラムを、各モジュールに対応してゲート回路 42 a 間の論理構成を変化させてハードウェア的に実行することができ、従来の C P U を用いた演算システムに比べて高速で演算を実行することができる。

【0041】また、F P G A データモジュール 21 ~ 2 n のうちの少なくとも 1 のモジュールが他のモジュールを呼び出すためのデータを含んでいるが、このような他のモジュールの呼び出しを含む F P G A データモジュールが F P G A データメモリ 41 にロードされた場合に、これを呼び出し検出部 43 が検出している。そして、この検出結果に基づいて、退避スタック 44 へのゲートアレイ 42 の内部状態（フリップフロップ 42 b が保持するデータ）の退避、引数受け渡し部 45 を介した引数の受け渡しを行っている。また、呼び出し先のモジュールに従った演算が終了したときに、退避スタック 44 に退避した内部状態の復元、引数受け渡し部 45 を介した呼び出し元のモジュールへの引数の受け渡しを行っている。このような仕組みを備えることによって、モジュールの呼び出しを含む大規模なプログラムをハードウェア的に実行することが可能となる。

【0042】また、呼び出し検出部 43 が他のモジュールの呼び出しを検出したときに、ゲートアレイ 42 の内部状態（フリップフロップ 42 b が保持するデータ）を退避するのは、先入れ後出し方式の退避スタックである。このため、他のモジュールから呼び出されたモジュールがさらに他のモジュールを呼び出すようなプログラムを実行することもできる。さらに、実行中のモジュールが自身を呼び出す再帰型のプログラムを実行することもできる。

【0043】さらに、F P G A データモジュール 21 ~ 2 n は、モジュール分割されたソースプログラム 51 ~ 5 n をそれぞれコンパイラ 6 によってコンパイルしたものである。以上のような特徴を有することによって、この演算システムにおいて実行すべきプログラムは、モジュール別にソースプログラムの開発を進めたり、ソースプログラムの各モジュールを部品として利用したりすることが可能となり、その開発期間を短縮することができる。

【0044】本発明は、上記の実施の形態に限られず、種々の変形、応用が可能である。以下、本発明に適用可能な上記の実施の形態の変形態様について説明する。

【0045】上記の実施の形態では、ロード 3 は、F P G A データ記憶部 2 に記憶されたいずれかの F P G A データモジュール 21 ~ 2 n を、そのまま F P G A データメモリ 41 にロードするものとしていた。これに対して、F P G A データモジュール 21 ~ 2 n がマクロを含み、F P G A データ記憶部 2 にマクロデータを記憶させておき、ロード 3 が F P G A データメモリ 41 にロードする際に、マクロ展開をするものとしてもよい。

【0046】上記の実施の形態では、ソースプログラム 51 ~ 5 n をそれぞれコンパイルした F P G A データモジュール 21 ~ 2 n を、F P G A デバイス 4 の F P G A データメモリ 41 に適時ロードしていくものとしていた。これに対して、ソースプログラム 51 ~ 5 n をそのままロードするようにした演算システムを構成することもできる。図 2 は、このような場合の演算システムの構成を示す。

【0047】この演算システムでは、ロード 3' は、制御部 46' からの指示に基づいて、プログラム記憶部 5 に記憶されたモジュール別のソースプログラム 51 ~ 5 n を適時メモリ 41' にロードする。インタプリタ 47 は、メモリ 41' にロードされたソースプログラム中の命令を 1 命令ずつ順次解釈し、その解釈結果に従ってゲートアレイ 42' を構成するゲート回路 42 a に論理構成を行わせるべく所定の信号を出力する。解釈の結果、他のモジュールのソースプログラムを呼び出す命令であった場合には、その旨を制御部 46' に通知する。

【0048】制御部 46' は、他のモジュールの呼び出しが通知されると、ゲートアレイ 42' の内部状態（フリップフロップ 42 b に保持されているデータ）と、呼

び出し元のソースプログラムのモジュールを識別するためのデータと、次に実行をすべき命令を示すデータを退避スタック 44 に退避すると共に、フリップフロップ 42b に保持されているデータのうち呼び出し先のモジュールに引数として渡すものを、引数受け渡し部 45 に一時保存させる。そして、ロード 3' に呼び出し先のソースプログラム 51 ~ 5n をロードさせ、引数受け渡し部 45 に一時保存されたデータを入力データとしてゲートアレイ 42' に与える。

【0049】また、呼び出し先のソースプログラムに従った演算が終了すると、ゲートアレイ 42' からの出力データを呼び出し元のモジュールに渡す引数として引数受け渡し部 45 に一時保存させる。そして、退避スタック 44 に退避されたデータに従って呼び出し元のソースプログラムを再びメモリ 41' にロードさせ、退避スタック 44 に退避された内部状態をフリップフロップ 42b に戻し、引数受け渡し部 45 に一時保存された引数をフリップフロップ 42b のうちの所定のものに書き込ませる。そして、退避スタック 44 に退避されたデータに基づいて呼び出し元のモジュールのソースプログラムに従った演算を再開させる。

【0050】なお、インタプリタ 47 は、複数のゲート回路の組み合わせによるハードウェアで構成することができ、その出力によってゲートアレイ 42' に含まれるゲート回路 42a の論理構成を、演算の実行速度にほとんど影響を与えることなく高速に行うことができる。また、ここでのゲートアレイ 42' は、ソースプログラム中の各命令を終了したときのデータをフリップフロップ 42b の所定のものに保持させることで、各命令を順次実行していくことができる。

【0051】以上のようにインタプリタ 47 を含む構成とすることによって、ソースプログラム 51 ~ 5n をモジュール別に順次 FPGA デバイス 4' にロードしていくことが可能となる。このため、FPGA デバイス 4' の構成に合わせたコンパイラがなくても、複数のモジュールからなる大規模なプログラムに従った演算を、ハードウェア的に高速に行うことが可能となる。

【0052】また、この実施の形態の演算システムを互いに連結可能な構成として、並列処理や分岐処理を、互いに連結された複数の演算システムが分担して行うようにしてもよい。具体的には、この演算システムは、たとえば、図 3 に演算システム 1A として示す構成を有していてもよい。

【0053】図示するように、演算システム 1A は、図 1 に示す演算システム 1 と実質的に同一の構成を備え、更に、補助演算制御部 7 を備えるものとする。補助演算制御部 7 は論理回路等より構成されており、他の演算システム（たとえば、図 1 あるいは図 3 に示す構成を有する演算システム）のロード 3、ゲートアレイ 42 及び引数受け渡し部 45 に着脱可能に接続され、後述する動作

を行う。

【0054】なお、複数の他の演算システムが演算システム 1A に接続されてもよい。具体的には、たとえば図 4 に示すように、演算システム 1B 及び 1C のそれぞれのロード 3、ゲートアレイ 42 及び変数引き渡し部 45 が、演算システム 1A の補助演算制御部 7 に接続されていてもよい。なお、演算システム 1B 及び 1C は、たとえば、図 1 あるいは図 3 に示す構成と実質的に同一の構成を有したものであればよい。ただし、FPGA データ記憶部 2 を必ずしも備えていなくてもよい。

【0055】図 3 の演算システム 1A は、図 1 の演算システム 1 と実質的に同一の動作を行う。そして、自己の FPGA データメモリ 41 にロードされた FPGA データモジュールに、他の演算システムに実行させるべき FPGA データモジュールを呼び出すデータが含まれていると、自己に接続された他の演算システムにこの FPGA データモジュールをロードさせ、演算を行わせて、演算結果を取得する。

【0056】以下、演算システム 1A が、図 4 の演算システム 1B 及び 1C に並列処理を行わせる動作を例として、演算システム 1A が自己に接続された他の演算システムに FPGA データモジュールをロードさせ、演算を行わせて演算結果を取得する動作を説明する。なお、以下では、FPGA データモジュール 21 が最初にロードされるものとし、FPGA データモジュール 21 は、FPGA データモジュール 2x を呼び出し、演算システム 1A は、演算システム 1B 及び 1C に FPGA データモジュール 2x をロードさせるものとする。

【0057】FPGA データモジュール 2 が演算システム 1A の FPGA データメモリ 41 にロードされると、演算システム 1A のゲート回路 42a が論理構成される。そして、演算システム 1A のゲートアレイ 42 に外部からの入力データが入力されると、FPGA データモジュール 21 に応じた演算が演算システム 1A のゲートアレイ 42 において実行される。

【0058】一方、演算システム 1A の呼び出し検出部 43 は、FPGA データメモリ 41 にロードされた FPGA データモジュール 21 に、演算システム 1B 及び 1C にロードさせるべき FPGA データモジュール 2x を呼び出すためのデータが含まれていることを検出し、その旨を制御部 46 に通知する。

【0059】その後、演算システム 1A の制御部 46 は、演算システム 1A のロード 3 を制御し、呼び出し先である FPGA データモジュール 2x を演算システム 1A の FPGA データメモリ 41 にロードさせる。FPGA データモジュール 2x がロードされると、演算システム 1A のゲートアレイ 42 は、この FPGA データモジュール 2x を取得する。そして、FPGA データモジュール 21 に応じた処理の一環として、この FPGA データモジュール 2x を演算システム 1A の補助演算制御部

7に供給し、演算を停止する。

【0060】また、演算システム1Aの制御部46は、演算システム1Aのフリップフロップ42bに保持されているデータのうちでFPGAデータモジュール2xに引数として渡すデータ（演算システム1Bに供給するデータ、及び、演算システム1Cに供給するデータ）を、演算システム1Aの補助演算制御部7に供給する。

【0061】演算システム1Aの補助演算制御部7は、演算システム1B及び1Cのローダ3を制御し、FPGAデータモジュール2xを、演算システム1B及び1CのFPGAデータメモリ41にそれぞれロードさせる。この結果、演算システム1B及び1CにFPGAデータモジュール2xがロードされ、演算システム1B及び1Cのゲート回路42aが論理構成される。

【0062】次いで、演算システム1Aの補助演算制御部7は、演算システム1Aの制御部46より引数として供給されたデータのうち、演算システム1Bに供給すべきものを、入力データとして演算システム1Bのゲートアレイ42に入力し、演算システム1Cに供給すべきものを、入力データとして演算システム1Cのゲートアレイ42に入力する。この結果、演算システム1B及び1Cのゲートアレイは、FPGAデータモジュール2xに応じた演算を、各自に供給されたデータが表す引数を与えられたものとして実行する。

【0063】FPGAデータモジュール2xに応じた演算が終了すると、演算システム1B（又は1C）の制御部46は、演算システム1B（又は1C）のゲートアレイ42からの出力データを、呼び出し元のFPGAデータモジュール21に渡す引数として、演算システム1B（又は1C）の引数受け渡し部45に一時保存させる。

【0064】演算システム1Aの補助演算制御部7は、演算システム1B及び1Cの引数受け渡し部45に出力データが一時保存されたことを検知し、これらの出力データを、演算システム1B及び1Cの引数受け渡し部45より取得する。そして、取得した各出力データを、演算システム1Aのフリップフロップ42bの所定のもの書き込む。この状態で、演算システム1Aのゲートアレイ42は、FPGAデータモジュール21に従った演算を再開する。この結果、最終的な演算結果が出力データとして出力される。

【0065】この発明の実施の形態の演算システムが図3に示す構成を有していれば、単一の演算システムでは短時間で完了できない演算や、並列処理を要する演算も、必要に応じて演算システムを追加することにより、短時間で完了させることが可能となる。

【0066】また、演算システム1Aに接続される他の

演算システムが図3に示す構成を有している場合、当該他の演算システムは、自己の補助演算制御部7に接続された演算システムにFPGAデータモジュールをロードさせ、演算を行わせて演算結果を取得することが可能である。従って演算の手順を柔軟に構成することが可能である。

【0067】なお、演算システム1Aが自己に接続された他の演算システムにソースプログラムをロードさせ、演算を行わせて演算結果を取得するようにしてもよい。ただし、この場合、演算システム1Aに接続される他の演算システムは、たとえば図2に示す構成を有しているものとする。

【0068】

【発明の効果】以上説明したように本発明によれば、複数のプログラムモジュールからなる大規模なプログラムであっても、各プログラムモジュールを適時メモリにロードしていく仕組みを有するので、該プログラムに応じた演算の実行をハードウェアで実現することが可能となる。

【図面の簡単な説明】

【図1】本発明の実施の形態にかかる演算システムの構成を示すブロック図である。

【図2】本発明の他の実施の形態にかかる演算システムの構成を示すブロック図である。

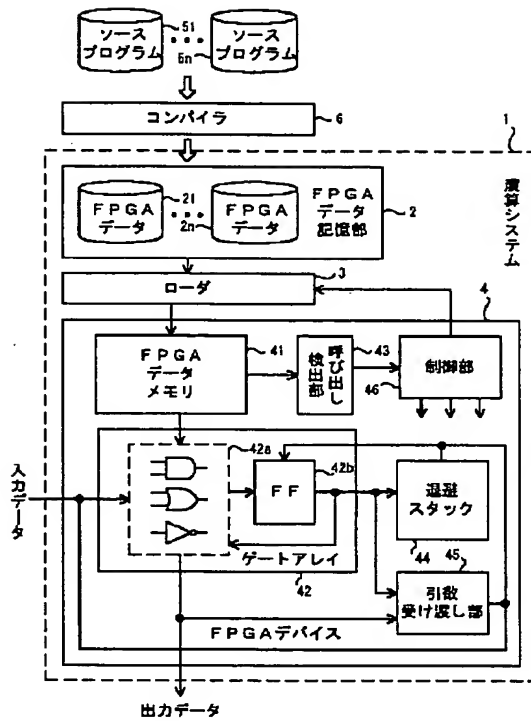
【図3】本発明の他の実施の形態にかかる演算システムの構成を示すブロック図である。

【図4】本発明の実施の形態にかかる演算システムが複数連結されて用いられる場合の構成を示すブロック図である。

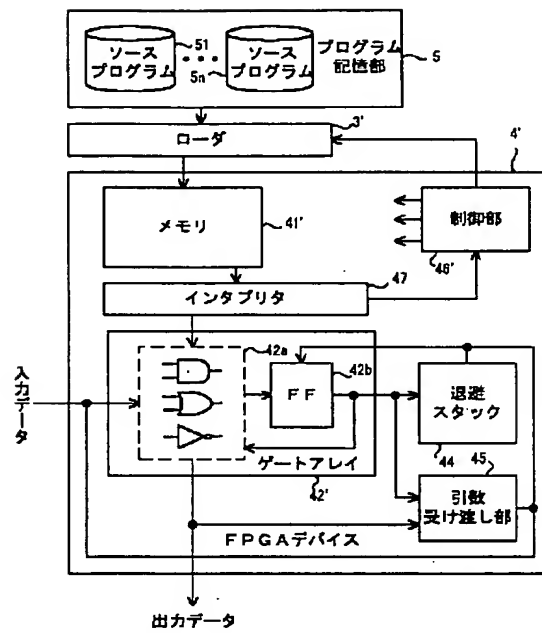
【符号の説明】

- 1、1A、1B、1C 演算システム
- 2 FPGAデータ記憶部
- 3 ローダ
- 4 FPGAデバイス
- 6 コンパイラ
- 7 補助演算制御部
- 21～2n、2x FPGAデータモジュール
- 41 FPGAデータメモリ
- 42 ゲートアレイ
- 42a ゲート回路
- 42b フリップフロップ
- 43 呼び出し検出部
- 44 退避スタック
- 45 引数受け渡し部
- 46 制御部
- 51～5n ソースプログラム

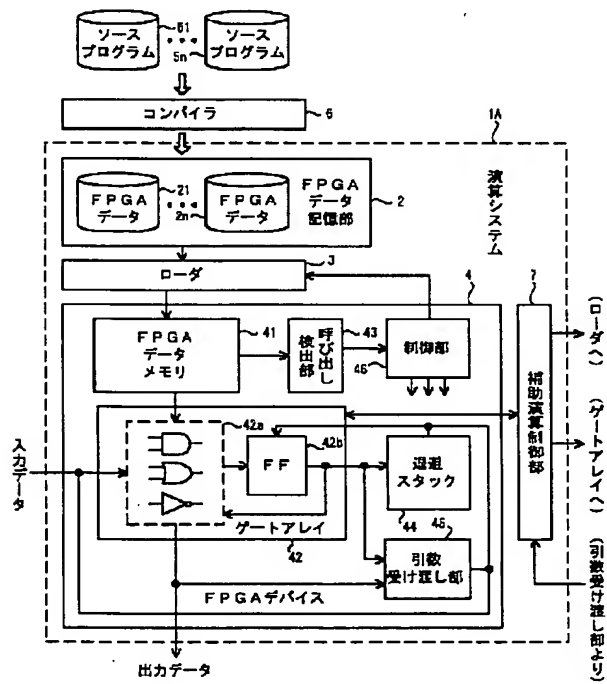
【図1】



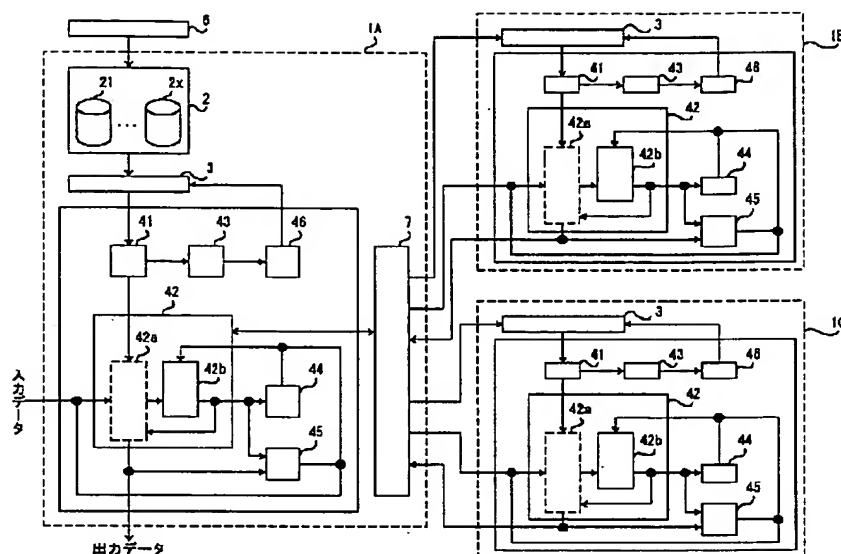
【図2】



【図3】



【図 4】



フロントページの続き

(72)発明者 長谷部 鉄也
 神奈川県横浜市都筑区東方町1番地 東京
 エレクトロンデバイス株式会社内
 (72)発明者 林 博昭
 神奈川県横浜市都筑区東方町1番地 東京
 エレクトロンデバイス株式会社内

(72)発明者 三田 高司
 神奈川県横浜市都筑区東方町1番地 東京
 エレクトロンデバイス株式会社内
 Fターム(参考) 5B033 DE08
 5B076 AA07 BA00
 5J042 BA01 CA15 CA20 CA22 CA23
 CA27 DA00